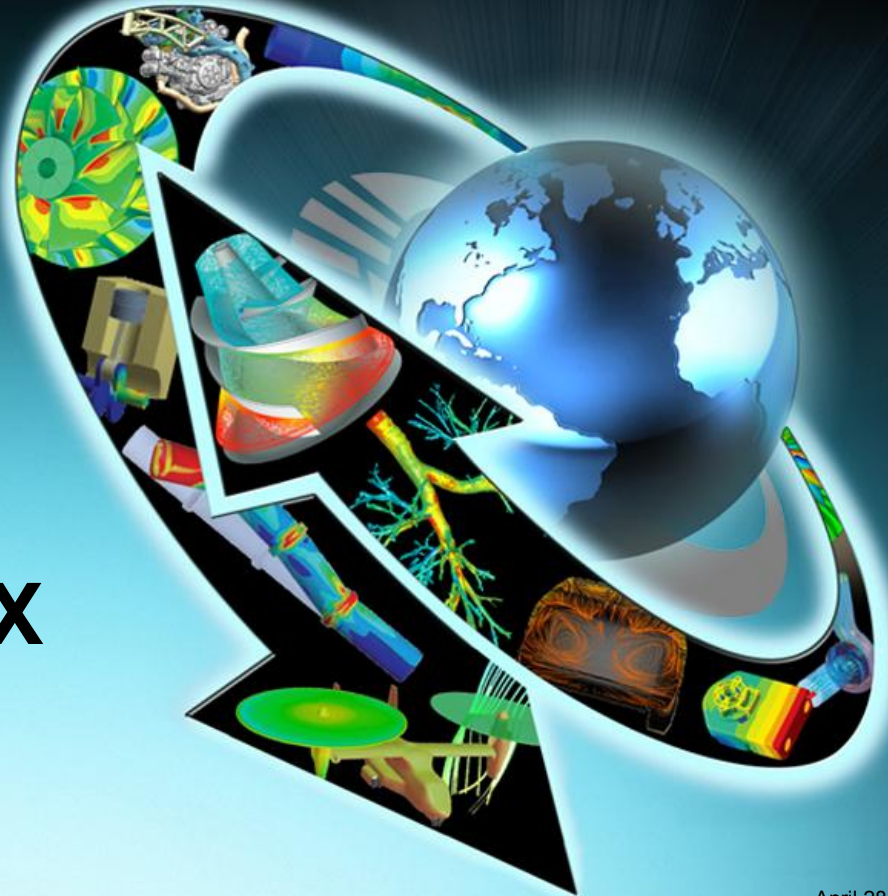




## Workshop 8

## Transient Brake Rotor

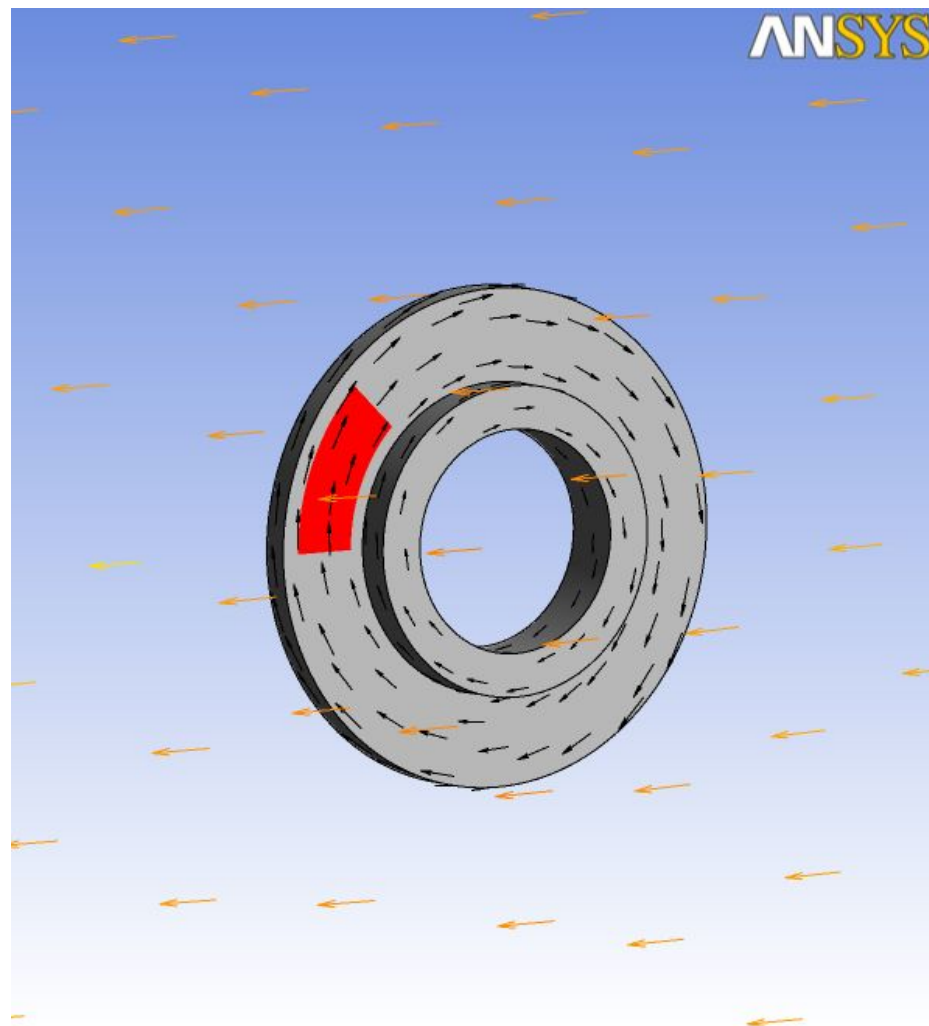
## Introduction to CFX



# Transient Brake Rotor

This case models the transient heating of a steel rear disk brake rotor on a car as it brakes from 60 to 0 mph in 3.6 seconds.

To keep solution times to a minimum the case has been simplified by removing the wheel and brake assembly to leave only the brake rotor. The brake pad is modeled by applying a heat source to a small region of the brake rotor.



# Assumptions

- The ambient air temperature is 81 F and the rotor is at ambient temperature before braking begins
- The vehicle tire size is 205/55/R16
- The total vehicle weight including passengers and cargo is 1609 kg
- The entire kinetic energy of the vehicle is dissipated through the brake rotors
- Energy dissipation during braking is split 70/30 between the front and rear brakes and split evenly between the left and right sides
- The vehicles speed reduces linearly from 60 to 0 mph in 3.6 seconds

# Solution Approach

- The solution is transient, so you will need to begin by solving a steady-state case at a vehicle speed of 60 mph



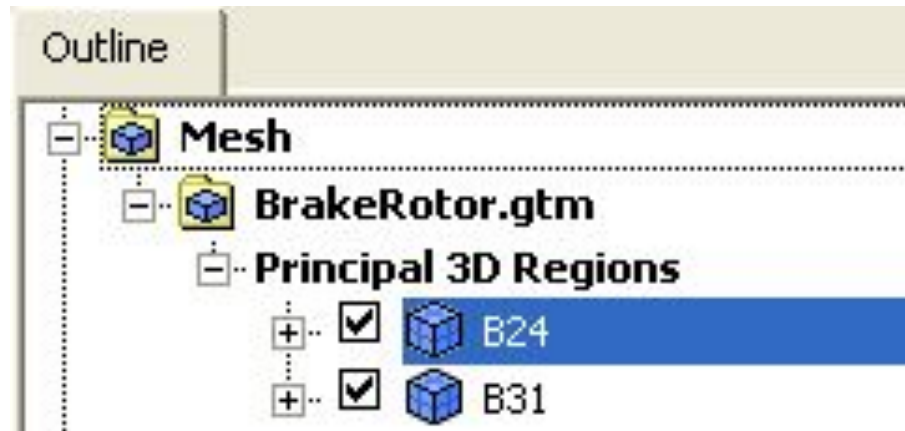
Transient simulations usually need to begin from a converged steady-state simulation. This establishes the initial fluid field so that the transient solution can start smoothly

- You will need two domains; a solid domain for the brake rotor and a fluid domain for the surrounding air
- The reference frame will be that of the vehicle. So the rotor will be spinning relative to this reference frame and air will be flowing past at the vehicle velocity

## Start Steady-State Simulation

1. Start CFX-Pre in a new working directory and create a new simulation named *BrakeDisk*
2. Right-click on *Mesh* in the *Outline* tree and import the CFX-Mesh file named *BrakeRotor.gtm*
  - The rotor mesh will be imported along with a bounding box surrounding the rotor
3. In the *Outline* tree, expand *Mesh > BrakeRotor.gtm > Principal 3D Regions*
  - There are two 3D regions in this mesh named B24 and B31

# Examine Mesh Regions




## 4. Click once in the tree on each of these 3D regions

- The mesh bounding each 3D region is displayed in the Viewer
- Notice that a mesh exists for the solid brake rotor and for the surrounding fluid region. These meshes are in separate 3D regions but still within the same Assembly

# Create the Fluid Domain

By default the *Simulation Type* is set to *Steady-State*, so the next step is to create the fluid domain

1. Select the Domain icon from the toolbar  and enter the *Name* as *AirDomain*
2. Pick the *Location* corresponding to the air region from the drop-down menu
  - The regions are highlighted in the Viewer to assist you
3. The fluid domain uses *Air Ideal Gas* as the working fluid at a *Reference Pressure* of 1 [atm]; the domain is *Stationary* relative to the chosen reference frame and *Buoyancy* (gravity) can be neglected. Use this information to set appropriate Basic Settings for this domain



# Create the Fluid Domain

4. Switch to the *Fluid Models* tab for the domain
5. Set the *Heat Transfer Option* to *Thermal Energy* and leave the *Turbulence Option* set to the default *k-Epsilon* model
6. Switch to the *Initialisation* tab for the domain



Initialisation must be set separately for each domain when both fluid and solid domains are included in a simulation. You cannot set global initial condition because some quantities do not apply in solid domains (e.g. velocity, pressure)



# Create the Fluid Domain

7. Enable the *Domain Initialisation*, toggle
  - All settings can then be left at their default values
8. Click *OK* to create the domain

Details of **AirDomain** in **Flow Analysis 1**

Basic Settings | Fluid Models | Initialisation | Solver Control

☒ Domain Initialisation

☐ Coord Frame

Initial Conditions

Velocity Type: Cartesian

Cartesian Velocity Components

Option: Automatic

☐ Velocity Scale

Static Pressure

Option: Automatic

Temperature

Option: Automatic

Turbulence

Option: Medium Intensity and Eddy V

# Create the Solid Domain

The next step is to create the solid domain for the brake rotor.

1. Create a new domain named *Rotor*
2. Pick the *Location* corresponding to the brake rotor
3. Set the *Domain Type* to *Solid Domain*
4. Set the *Material* to *Steel*
5. Leave the *Domain Motion Option* as *Stationary*



For this case it is not necessary to solve the solid domain in a rotating reference frame. It is easier to leave it in a stationary reference frame, then define Solid Motion on the next tab. See the notes at the end of this workshop for more details.

1. Switch to the *Solid Models* tab and enable the *Solid Motion* toggle

# Create Expressions

## 7. Set the *Solid Motion Option* to *Rotating*

The next quantity to enter is the *Angular Velocity*. This needs to be calculated based on the vehicle speed (60 mph) and the radius of the tire attached to the brake rotor. The tires were specified as 205/55/R16 (205 mm tire width, aspect ratio of 55, 16" rim diameter). Next you will create expressions to calculate the Angular Velocity.

## 8. Switch to the *Outline* tab (do not close the *Domain* tab)

## 9. Right-click on *Expressions* in the tree and select *Insert > Expression*

- You may need to expand the *Expressions, Functions and Variables* entry in the tree to be able to right-click on *Expressions*

## 10. Enter the expression *Name* as *Speed* and click *OK*

- The *Expressions* tab will appear

# Create Expressions

11. In the *Definition* window (bottom-left of the screen) enter  $60 [\text{mile hr}^{-1}]$  then click *Apply*
12. Right-click in the top half of the *Expressions* window and select *Insert > Expression*; enter the *Name* as *TireRadius*
13. Enter the *Definition* as  $(16 [\text{in}] / 2) + (205 [\text{mm}] * 0.55)$  and click *Apply*



Notice that you do not need to convert between different units; just provide units when defining quantities and CFX will convert when necessary

11. Create another expression named *Omega*, type the *Definition* as  $\text{Speed} / \text{TireRadius}$  and then click *Apply*
12. Now switch back to the *Domain: Rotor* tab

# Complete the Solid Domain

16. Click the expression icon next to the *Angular Velocity* field and type in *Omega* (the name of the expression you just created)
17. Pick the *Rotation Axis* as the *Global X* axis
18. On the *Initialisation* tab set the *Temperature Option* to *Automatic with Value* and enter a *Temperature* of *81 [ F ]*
  - Make sure you have changed the units to *F*
19. Now click *OK* to create the domain

# Create Boundary Conditions

Boundary conditions are needed for the bounding box of the air domain. You will create an inlet boundary upstream of the rotor, an outlet boundary downstream of the rotor and an opening boundary for the remaining bounding surfaces. Start with the inlet boundary:

1. In the *Outline* tree, right-click on *AirDomain* and select *Insert > Boundary*. Enter the *Name* as *AirIn* when prompted and click *OK*
2. On the *Basic Settings* tab, set the *Boundary Type* to *Inlet* and the *Location* to *Inlet*
3. On the *Boundary Details* tab, set the *Mass And Momentum Option* to *Normal Speed*
4. In the *Normal Speed* field click the expression icon and enter *Speed*
  - This is one of the expressions you created earlier

# Create Boundary Conditions

5. Set the *Heat Transfer Option* to *Static Temperature* and enter the a value of  $81 [F]$
6. Click *OK* to create the inlet boundary

Now create the outlet boundary condition:

1. Right-click on *AirDomain* and insert a boundary named *AirOut*
2. Use the following setting for this boundary:
  - *Boundary Type* = *Outlet*
  - *Location* = *Outlet*
  - *Mass And Momentum Option* = *Average Static Pressure*
  - *Relative Pressure* =  $0 [Pa]$
3. Click *OK* to create the outlet boundary



# Create Boundary Conditions

Lastly, create the opening boundary condition:

1. Insert a boundary named *AirOpening* into the *AirDomain*
2. Use the following settings for this boundary:
  - *Boundary Type = Opening*
  - *Location = OuterWalls*
  - *Mass And Momentum Option = Entrainment*
  - *Relative Pressure = 0 [ Pa ]*
  - *Turbulence Option = Zero Gradient*
  - *Heat Transfer Option = Opening Temperature*
  - *Opening Temperature = 81 [ F ]*
3. Click *OK* to create the opening boundary

# Create Domain Interface

Domain Interfaces are required when more than one domain exists in your simulation. Without domain interfaces one domain would not see or feel the effect of neighboring domains. A Default Fluid Solid Interface should already exist, but we will manually create the interface here as a practice exercise.

1. Select the Domain Interface icon from the toolbar and enter the *Name* as *RotorInterface*
2. Set the *Interface Type* to *Fluid Solid*
3. For *Interface Side 1*, set the *Domain (Filter)* to *AirDomain*; pick both *BrakePadsFluidSide* and *RotorFluidSide* from the *Region List*



The *Domain (Filter)* is only used to limit the *Region List* to regions in the selected domain. You do not have to use the filter, but it makes region picking easier and less error prone

# Create Domain Interfaces



The regions *BrakePadsFluidSide* and *RotorFluidSide* were created when the mesh was generated. By considering what regions will be needed at the mesh generation stage, the set up in CFX-Pre is made easier

4. For *Interface Side 2*, set the *Domain (Filter)* to *Rotor*. Pick *BrakePadsSolidSide* and *RotorSolidSide* from the *Region List*
5. Under *Interface Models*, leave the *Frame Change* and *Pitch Change Option* set to *None*



See the notes at the end of this workshop for more details on appropriate Frame Change models for Fluid Solid Interfaces

4. Click OK to create the Domain Interface
  - Notice that the default interface no longer exists

# Modify Interface Boundaries

Notice in the *Outline* tree that new *Side 1* and *Side 2* boundary conditions have been created automatically in the Air and Solid domains. These boundary conditions are associated with the Domain Interface

1. Double click *RotorInterface Side 1* in the *AirDomain*
2. Select the *Boundary Details* tab

By default the boundary condition is a no slip, stationary, smooth wall. It is necessary to modify these settings so that the air feels a rotating wall at the fluid solid interface



Boundary Conditions are *always* relative to the *local* frame of reference for the domain. In this case the reference frame for both domains is stationary, so we need to add a wall velocity to the fluid side.

## Modify Interface Boundaries

3. Enable the *Wall Velocity* toggle
4. Set the *Option* to *Rotating Wall*
5. Set the *Angular Velocity* to the expression *Omega*
6. Pick *Global X* as the *Rotation Axis*
7. Click *OK*



# Set Solver Controls

The last step before running the steady-state solution is to set the Solver Control parameters. Default Solver Control parameters already exist, so you can edit the existing object:

1. Double-click the *Solver Control* entry in the *Outline* tree
2. Change the *Fluid Timescale Control* to *Physical Timescale*
  - Based on the domain length (about 1.2 [m]) and the inlet velocity (60 mph), the advection time for air through the domain is about 0.045 [s]
3. Set the *Physical Timescale* to 0.02 [s]
4. Set the *Solid Timescale Control* to *Physical Timescale*
5. Set the *Solid Timescale* to 100 [s]
6. Click *OK*

# Run the Steady-State Solution

You can now run the case in the Solver

1. Select the *Run Solver and Monitor* icon 
2. Click *Save* to write the *BrakeDisk.def* file and launch the *Solver Manager*
  - The solution should converge in about 60 iterations
3. When the Solver finishes, check the Domain Imbalance values in the out file
  - All imbalances should be well below 1%
4. Click the Post Process Results icon from the toolbar 



# Post-Processing

Since this case is just the starting point for the transient simulation, there is very little post-processing to perform.

1. Check that the solution looks correct by plotting velocity
2. On the *Variables* tab, double click on the *Temperature* variable. Check that the *Min* and *Max* values are almost identical
3. Quit CFX-Post and return to the *BrakeDisk* simulation in CFX-Pre
4. Save the CFX-Pre simulation

# Start Transient Simulation

Next you will define the transient simulation by modifying the steady-state simulation in CFX-Pre. Start by saving the simulation under a new name so that you do not overwrite the previous set up

1. Select *File > Save Case As...*
2. Enter the *File name* as *BrakeDiskTrn.cfx* and click *Save*

To set up the transient simulation you will need to:

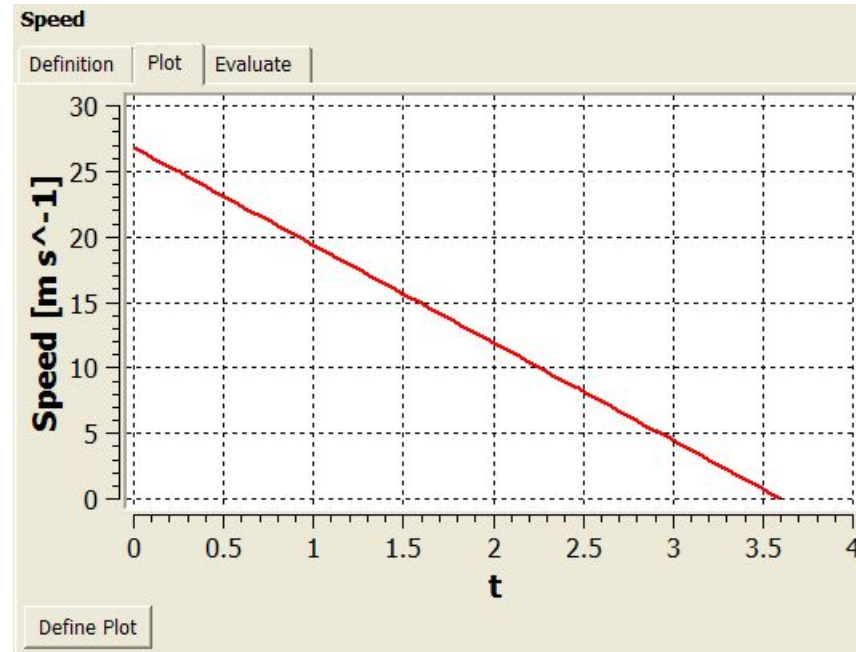
- Edit the expression for Speed so that the inlet velocity reduces with time
- Change the Simulation Type to Transient and enter the transient time step information
- Add a heat source to the braking surfaces to simulate the heat generated through braking. You'll need additional expressions for this
- Modify the Solver Controls
- Add some Monitor Points

# Edit Expressions

Start by defining the stopping time for the vehicle and then editing the expression for Speed based on the stopping time

1. Right-click on *Expressions* in the *Outline* tree, select *Insert > Expression* and enter the name as *StoppingTime*
2. Set the Definition to *3.6 [s]* and click *Apply*
3. Change the expression *Speed* to:  
$$60 [\text{mile hr}^{-1}] - (60 [\text{mile hr}^{-1}] / \text{StoppingTime}) * t$$
then click *Apply*
4. On the *Plot* tab, check the box for *t* and enter a range from 0 – 3.6 [s]
5. Click *Plot Expression*
  - You should see Speed decreasing linearly from about 27 to 0 [m s<sup>-1</sup>] as shown on the next slide

## Edit Expressions



6. Create a new expression named *Deltat* with a value of 0.05 [s]
- This expression will be used next to set the timestep size for the transient simulation

# Change Simulation Type

Next you will change the Simulation Type to Transient and enter information about the duration of the simulation

1. In the *Outline* tree, double click on *Analysis Type*
2. Set the *Analysis Type Option* to *Transient*
3. Enter the *Total Time* as the expression *StoppingTime*
4. Enter *Timesteps* as the expression *Deltat*
5. Set the *Initial Time Option* to *Automatic with Value* and use a *Time* of 0 [s]
  - Transient timesteps of 0.05 [s] will be taken, starting at 0 [s] and ending at 3.6 [s] for a total of 72 timesteps
6. Click *OK*

# Add a Braking Heat Source

To add a heat source to simulate the heat generated through braking, edit the solid side boundary condition associated with the interface *RotorInterface*. Notice that the interface covers the entire surface of the rotor, but a mesh region exists where the brake pads are located. In the *Outline* tree you can expand *Mesh* > *BrakeRotor.gtm* > *Principle 3D Regions* > *B31* > *Principle 2D Regions* to see the region *BrakePadsSolidSide*.

1. Edit the *RotorInterface Side 2* boundary condition in the *Rotor* domain
2. On the *Sources* tab enable the *Boundary Source* toggle, then the *Source* toggle and then the *Energy* toggle

## Add a Braking Heat Source

Using the assumptions listed at the start of the workshop, the energy to apply to the brake surface can be calculated. The vehicle velocity as a function of time and the vehicle mass is known. Therefore the kinetic energy dissipated through the brakes over one timestep can be calculated. It is also known that 15% of the total energy is dissipated through each rear brake rotor.

3. Switch to the *Expressions* tab, or double click *Expressions* from the *Outline* tree if the tab is not already open
4. Create a new expression named *Mass* with a value of *1609 [kg]* and click *Apply*

To calculate the kinetic energy lost over one timestep you need to know the change in Speed over the timestep. You already have an expression for the Speed at the end of the timestep, so you need an expression for the Speed at the end of the previous timestep.



# Add a Braking Heat Source

5. Right click on the expression named *Speed* and select *Duplicate...* from the pop-up menu
  - *Copy of Speed* will be created
6. Right click on *Copy and Speed* and *Rename* it to *SpeedOld*
7. Edit the Definition for *SpeedOld* to read:  
$$60 [\text{mile hr}^{-1}] - (60 [\text{mile hr}^{-1}] / \text{StoppingTime}) * (t - \text{Deltat})$$
8. Create a new expression named *DeltaKE*. Enter the *Definition* as:  $0.5 * \text{Mass} * (\text{SpeedOld}^2 - \text{Speed}^2)$

15% of *DeltaKE* will be applied to the rotor. The energy source term will be applied as a flux which has units of  $[\text{J s}^{-1} \text{m}^{-2}]$ . Therefore you need to divide by the timestep size and the area of the brake pads to obtain the correct flux. Lastly, the source needs to be limited to just the brake pad region within the *RotorInterface Side 2* boundary condition.

# Add a Braking Heat Source

9. Create a new expression named *HeatFlux*. Enter the Definition as:

$$\text{inside()}@REGION:BrakePadsSolidSide * 0.15 * \Delta KE / (\text{area()}@ REGION:BrakePadsSolidSide * \Delta t)$$



The *inside()* function returns 1 when inside the specified region, otherwise it returns 0



The syntax *@REGION:Name* is used to refer to any locator in the mesh. This differs from the standard *@Name* syntax which is used to refer to a physics locator (e.g. a domain, boundary condition, subdomain etc.). You can right-click in the *Definition* section to access these names.

9. Switch back to the *Boundary* tab for *RotorInterface Side 2*
10. Set the *Energy Option* to *Flux*
11. Enter the expression *HeatFlux* for the *Flux* and click *OK*

# Modify Solver Controls

## 1. Edit the *Solver Control* object from the *Outline* tree

The default transient Solver Control settings use a maximum of 10 coefficient loops per timestep with a RMS residual target of  $1e-4$ . Fewer loops may be used if the residual target is met sooner. If the residual target is not met after 10 loops the solver will continue on to the next timestep regardless. It is therefore important to check you are converging to an acceptable level during a transient simulation.



Convergence in transient simulations can be improved by using more coefficient loops or by using a smaller timestep. It is generally better to use a smaller timestep with fewer coefficient loops.

## 1. The default settings are appropriate for this simulation. Click *OK*

# Monitor Points

Monitor Points are used to monitor variables at x, y, z coordinates or monitor the value of expressions as the solution progresses.



Monitor points should be used whenever possible to assist with judging convergence. For steady-state simulations monitor a quantity of interest and check that it has reached a steady value when the solver finishes. In transient simulations monitor points are often the easiest way to produce time history plots of a variable or expression

1. Edit the *Output Control* object from the *Outline* tree
2. On the *Monitor* tab enable the *Monitor Options* check box
3. In the *Monitor Points and Expressions* frame, click the New icon to create a new monitor point
4. Enter the *Name* as *AvgRotorT* and click *OK*

# Monitor Points

5. Change the *Option* to *Expression*
6. Enter the *Expression Value* as  
`volumeAve (Temperature) @Rotor`
  - This expression will return the average temperature of the rotor
7. Click the New icon to create a second monitor point named *BrakeSfcT*.
8. Make sure that *BrakeSfcT* is selected, change the *Option* to *Expression* and enter the expression below. You can right click on the Expression Value field instead of typing.  
`areaAve (Temperature) @REGION:BrakePadsSolidSide`
  - This expression will return the average temperature on the specified region
9. Click *Apply* to commit the *Output Control* settings


# Transient Results

By default results are only written at the end of the simulation. You need to create transient results files to be able to view the results at different time intervals.

1. Switch to the *Trn Results* tab in the *Output Control* window and click the *Create New* icon
2. Change the Option to *Selected Variables*
  - By selecting only the variables of interest the transient results files are kept small
3. In the *Output Variables List*, use the ... icon to select the variables *Temperature* and *Velocity* (use the Ctrl key to pick multiple variables)
4. Set the *Output Frequency Option* to *Timestep Interval*
5. Enter a *Timestep Interval* of 4 then click *OK*

# Start Solver

The transient simulation is now ready to proceed to the solver.

1. Click the *Define Run* icon from the toolbar 
  - This will launch the Solver Manager but will not start the run. We need to provide an Initial Values File before running the Solver
2. Click *Save* to write the file *BrakeDiskTrn.def*
  - A Physics Validation Summary will appear
3. Read the Physics Validation message and then read the warning it is referring to which is shown in the message window below the Viewer. Click Yes to continue.
4. When the Solver Manager opens enable the *Initial Values Specification* toggle and select the file *BrakeDisk\_001.res*. Click *Start Run*.



# Monitor Completed Run

The solution time for the transient simulation is significantly more than for the steady-state simulation. Results files are provided for the transient simulation to save time.

5. Click the *Stop* icon in the Solver Manager after a couple of timesteps have been completed
6. In the Solver Manager select *File > Monitor Finished Run*
7. Browse to the directory where the previously run transient files are located, select the *.res* file then click *Open*
  - On the User Points tab the time history plots for the two monitor points are shown.
8. Check that the residual plots and imbalances show reasonable convergence
9. Click the Post-Process Results icon to proceed to CFX-Post

# Post Processing

Next you will make a transient animation showing the evolution of temperature on the surface of the rotor.




1. Edit the *RotorInterface Side 2* object
2. Colour the object by *Temperature* using a *Global Range*

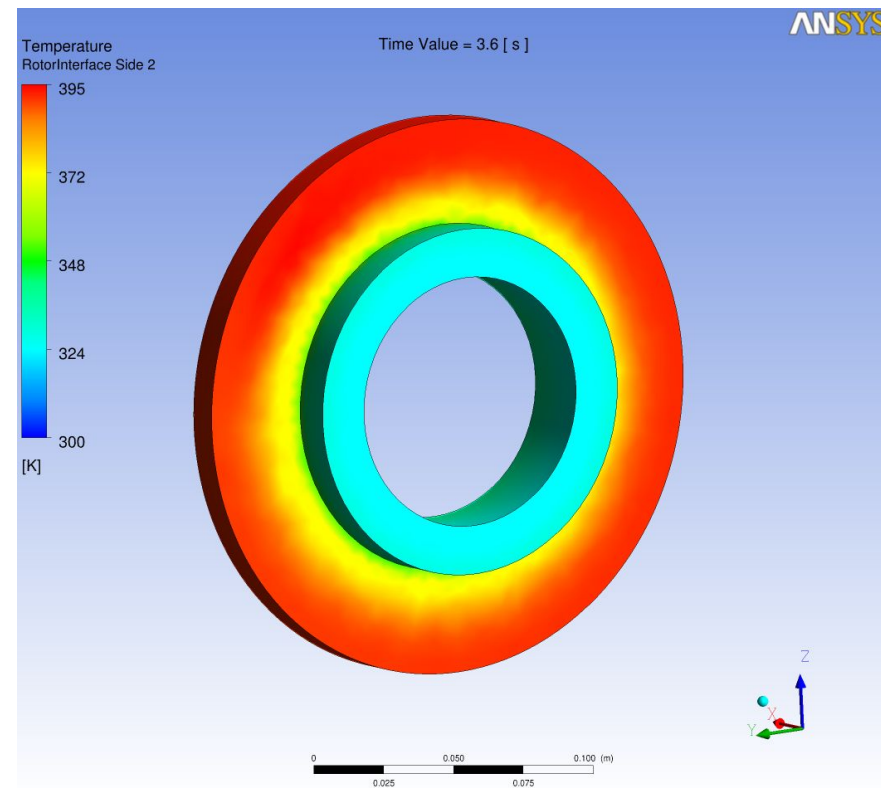


In transient simulations the global range of a variable covers all timesteps when the selected variable exists in the transient results files

1. Edit the *Default Legend View 1* object
2. On the *Appearance* tab, change the *Precision* to *0* and *Fixed* (the default is *3* and *Scientific*) and then click *Apply*
3. Orient the view similar to the image below

# Create Animation

6. Select the *Text* icon  from the toolbar then click *OK* to accept the default *Name*
7. On the *Definition* tab, enable the *Embed Auto Annotation* toggle
8. Set the *Type* to *Time Value* then click *Apply*
9. Select the *Animation* icon  from the toolbar
10. Select the *Quick Animation* toggle
11. Set the *Repeat* option to 1. You may need to turn off the *Repeat Forever* icon  first



# Create Animation

12. Enable the *Save Movie* toggle
13. Check that *Timesteps* is highlighted in the selection window and click the *Play* icon  to play and generate the animation
  - CFX-Post will generate one frame from each of the available transient results files. The animation file will be written to the current working directory.

# Rotating Solid Domains Notes

The following notes are for reference only and explain some of the features of rotating solid domains in greater depth.

In a solid domain both the *Domain Motion* and the *Solid Motion* can be set to *Rotating*. Setting the *Domain Motion Option* to *Rotating* for a solid domain in a transient simulation automatically includes the circumferential position for the solid domain in the results file. In other words, the solid domain will appear to rotate in the theta direction for visualisation purposes.

By itself, using *Domain Motion = Rotating* tells the solver to use mesh coordinates in the relative frame, similar to rotating fluid domains. It does not cause the solver to physically rotate the volumetric mesh or temperature field during the solution. Therefore the solution will look identical to that of a stationary solid domain.

# Rotating Solid Domains Notes

The reason for this behavior is not immediately obvious. However, there are many rotating solid cases that can be modeled as stationary solids, but for post-processing purposes you still want to see the solid rotate along with, say, the fluid domains to which it is connected. Turbomachinery blade cooling applications are a common example.

In some cases it is also necessary to account for the rotational motion of the solid energy, and the resulting temperature field. One of two approaches can be used to account for this effect, and the two are not exactly equivalent. Fortunately there is some flexibility in your choice of approach. Either approach is valid when you want energy to be distributed in the circumferential direction around the solid and the source of heat is stationary in the stationary frame.

# Rotating Solid Domains Notes

The first approach, as used in this workshop, is to use the *Solid Motion* settings on the *Domain > Solid Models* panel. The solid mesh is not physically rotated; instead a term is added to the solid energy equation to advect the energy using the defined velocity components or angular velocity. Therefore a stationary heat source applied to a solid boundary condition, like the brake pad for example, is felt throughout the entire disc rotor. Remember that we are in a stationary reference frame here, so the heat source applied to the boundary does not rotate.

The second approach is to account for the relative rotational motion at the Fluid-Solid interface using a rotating reference frame for the solid (*Domain Motion Option = Rotating*) combined with the *Transient Rotor Stator* (TRS) frame change model, leaving the *Solid Motion* undefined. The relative motion at the interface is accounted for by rotating the surface mesh at the interface. This modeling approach is appropriate in two situations: when the heat source is applied from the fluid side of the interface or when the heat source is applied from the solid side **and** the heat source rotates with the solid.

# Rotating Solid Domains Notes

As an example, if a hot jet of fluid is impinging on a cooler rotating solid, the entire rotating solid will heat up over time. If you do not use one of these two approaches then a single hot spot will form in the solid domain. In the first approach the *Domain Motion* is left as *Stationary* while the *Solid Motion* settings define the motion. The frame change model at the interface is left as *None* or *Frozen Rotor*. In the second approach there is no advection term in the solid energy equation (*Solid Motion* is not defined), but the mesh rotates at the interface (*Domain Motion* is *Rotating* and a TRS interface is used).

Note that in general you should not combine the two approaches. You would not use *Domain Motion* with *Transient Rotor Stator* and also define *Solid Motion* since this will rotate things twice.



# Rotating Solid Domains Notes

At the Fluid-Solid interface, Frame Change and Pitch Change options must be set. You should understand these concepts for Fluid-Fluid interfaces before understanding the following guidelines. The Fluid-Solid interface Pitch Change model can be *None*, *Automatic*, *Pitch Ratio* or *Specified Pitch Angles*. When the full 360 degree solid domain is modeled, as in this workshop, then *None*, *Pitch Ratio* of 1.0 and *Specified Pitch Angles* of 360 degrees on both sides are all equivalent options.

If you are modeling a periodic section of the fluid and solid domain, and a pitch change occurs at the interface, then you should use one of *Automatic*, *Pitch Ratio* or *Specified Pitch Angle* to correctly scale the heat flow profile across the interface, with the local magnitude scaled by the pitch ratio. In this case side 1 and side 2 heat flows should differ by the pitch ratio.

# Rotating Solid Domains Notes

Just as with rotating fluid domains, a rotating solid domain must be rotationally periodic or the full 360 degrees must be modeled. On the fluid side of the interface all Wall Velocities must be tangent to the rotating direction. Modeling a vented brake rotor, which has some walls moving normal to the rotating direction, would require a rotating solid domain, a rotating fluid domain surrounding the solid domain, and then a stationary fluid domain.